/**** Q1 Classe Flor *****/**

/* Na aula passada vimos a diferença entre *Valued Types* (tipos valorados) e *Referenced Types* (Tipos Referenciados). Abaixo temos uma classe (*Valued Type*) que mostra o princípio de funcionamento das referências. Veja código e faça um desenho para mostrar o que está ocorrendo */

```
class Flor {
   var cor : String = "brnaca"

   init (cor : String) {
      self.cor = cor
   }

   func show () {
      print (self.cor)
   }
}

print ("Inicialização")
var lirio = Flor (cor: "banca")
var azaleia = Flor (cor: "rosa")
lirio.show()
azaleia.show()
```

/* Você poderia criar uma struct chamada QeR (Questão e Resposta) com dois atributos String uma para pergunta e outro para resposta e tente fazer o mesmo experimento da classe Flor? qual o resultado? */

/**** Q2 Structs já possuem inicializa dores default para todos os atributos e classes possuem inicializa dores default apenas para objetos sem parâmetros *****/**

/* Compare a struct abaixo com a classe flor e veja que a struct não possui método init, mesmo assim compila (ver comentário no inicializador). Métodos init servem para fazer instâncias de tipo, ou seja, objetos de classe ou variáveis de structs*/

```
struct PontoR2{
  var x : Int
  var y : Int

  func show(){
    print("(\(self.x), \(self.y))")
  }
}
```

```
var px : PontoR2 = PontoR2(x: 10, y:35) // inicialização com atributos
print("Pontos(x,y)")
px.show()
```

/*refaça a struct QeR para possuir dois inicializadores um com pergunta com resposta e outro para perguntas sem resposta*/

/**** Q3 Métodos *****/**

/* Veja que na classe abaixo existem 3 métodos (func é a palavra-chave que os define). Veja que o código não executa por falta de implementação de um método. Você pode escrever esse método. */

```
class Counter {
   var count = 0 //atributo

   func show(){ // Pedir para fazer esse método
      print(self.count)
   }
   func increment() { //método
      count += 1
   }
   func increment(amount: Int) { //método
      count += amount
   }

}

let c1 =  Counter.init()
c1.increment(amount: 5) // Veja que aqui mudados o estado do atributo count
c1.increment(amount: 2) // logo mudamos o estado do objeto com o método increment
   c1.show()
   c1.reset() // zera o atributo count
   c1.show()
```

/* Agora que você fez a implementação do reset faça a implementação dos métodos de decremento (decrement). */

/* aqui temos um exemplo de um método que retorna o um valor (dobro de x). Veja onde foi colocado o valor do tipo de retorno e o a palavra reservada *return* */

```
func doubleX() -> Double {
   return self.x*2
}
```

*Você pode criar métodos de para retornar o dobro e o triplo de count? */*

Valeu pessoal, boa sorte!